User mannual

MiroMouse102

IEEE 国际标准电脑鼠

V1.2　　Date: 2007/08/13　　工程样机用户手册

类别	内容
关键词	电脑鼠、IEEE、走迷宫、机器人
摘　要	介绍 IEEE 电脑鼠 MiroMouse102 的硬件电路原理、软件设计方法等。

www.embedtools.cn
广州致远电子有限公司
产品用户手册

ZHIYUAN ELECTRONICS
V1.2

©2007 Guangzhou Zhiyuan Electronics CO., LTD

目　　录

1. 硬件结构 1

1.1 简介 1
1.2 硬件原理 2

2. 反射式红外线传感器 9

2.1 反射式红外线传感器设计方案 9
2.2 一体化红外接收头工作原理 10
2.3 检测障碍物的软件设计 10

3. 迷宫挡板检测 12

3.1 原理分析 12
3.2 调制信号产生 12
3.3 抗干扰处理 13
3.4 软件设计参考 14

4. 电机的调速 20

4.1 电机的调速 20
4.2 程序设计 21

5. MICROMOUSE 车速检测 24

5.1 车速检测程序设计 24

6. 使用 JTAG 引脚作 GPIO 26

6.1 使用方法 26

©2007 Guangzhou Zhiyuan Electronics CO., LTD

1. 硬件结构

1.1 简介

MicroMouse102 电脑老鼠，采用美国 LuminaryMicro 公司生产的 32 位 ARM CortexM3 处理器 LM3S102，控制和检测红外传感器；主 CPU 根据检测到的传感信号，控制电机驱动电路调整行走路径，直到到达终点。

©2007 Guangzhou Zhiyuan Electronics CO., LTD

1.2 硬件原理

MicroMouse102 电脑鼠原理图如图 1.1 所示。

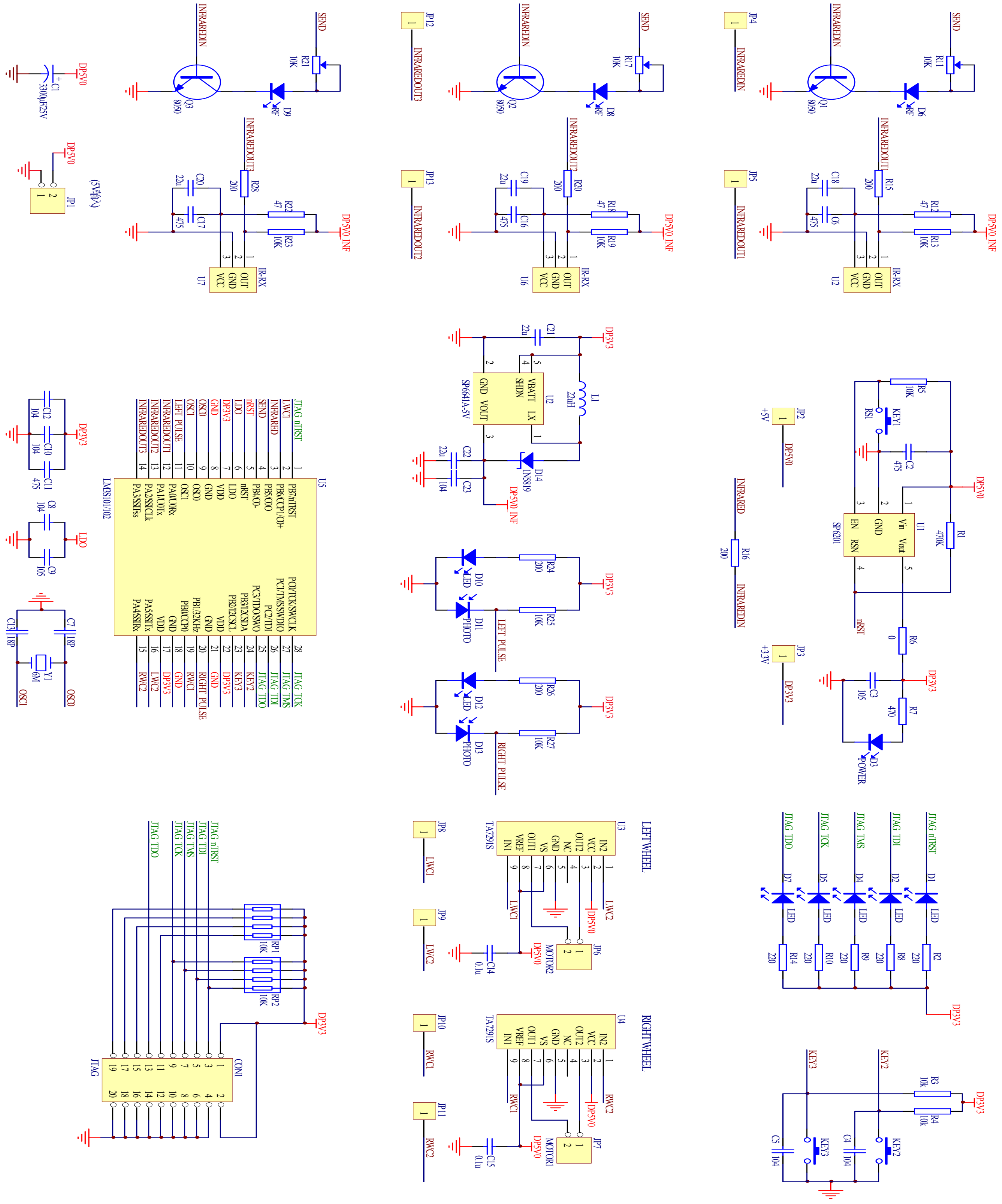

图 1.1 MicroMouse102 电脑鼠原理图

©2007 Guangzhou Zhiyuan Electronics CO., LTD

1.2.1 电源电路

电脑鼠的电源采用 4 节 1.5V 的 5 号电池供电，电源插座为 2.54-2T 型插座，如图 1.2 所示。

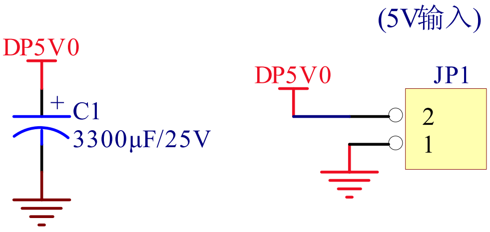

图 1.2 电源接口电路

1.2.2 LDO 和复位电路

LDO 和复位电路如图 1.3 所示。LDO 和复位电路采用 Sipex 公司性价比高的单芯片解决方案 SP6201EM5-3.3。5V 电源经过 SP6201 可以输出 3.3V 最大 200mA 电流给微处理器和用户外设供电。SP6201 还有使能输入和 Reset Not 输出引脚，通过这些引脚实现对微处理器的复位控制。

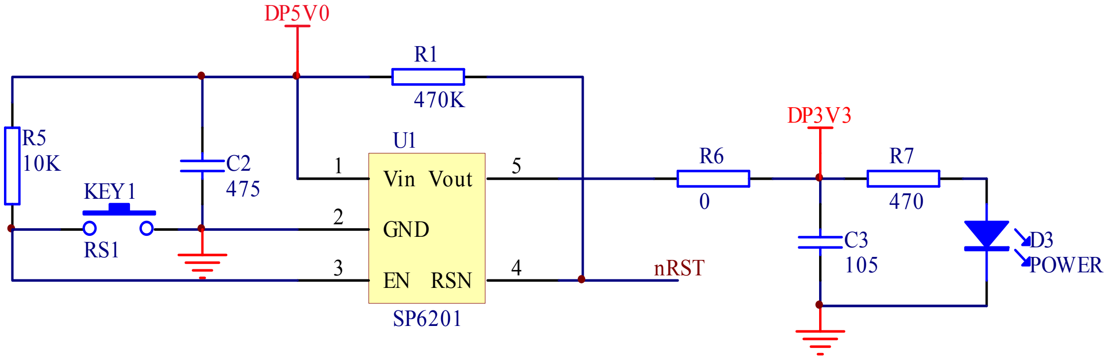

图 1.3 LDO 和复位电路

使能（非关断）输入：LDO 通过拉底 EN 管脚关闭，通过拉高 EN 管脚打开。如果不需要关断 LDO，EN（管脚 3）应连接到 IN（管脚 1），来保持调节器输出在任何时候都有效。使能阈值为 9V，在整个温度和 VIN 电压范围内的变化不能大于 100mV。使能阈值的变化总在 100mV 以内。关断电流保证小于 1uA，用户不需要总是将使能管脚拉底（0V）。标准的 TTL 或 CMOS 电平可使器件从完全运行到完全关闭。

Reset Not（V_{OUT} 良好）输出：所有固定输出的版本器件都提供一个 V_{OUT} 良好指示器，管脚 4（RSN）。这是一个开漏、逻辑输出管脚，当由 V_{OUT} 提供的电压超过规定调节范围的 4％时，它可用来使微处理器或微控制器保持复位。Reset Not 功能包含一个 1％的滞后，使得不会因为 LDO 的输出噪声而发出错误的警告。Reset Not 功能在 10～50us 内起作用。

1.2.3 升压电路

升压芯片采用 Sipex 公司的低静态电流、高效率的升压芯片 SP6641A，升压电路如图 1.4 所示。输入电压 3.3V，输出电压 5V。

©2007 Guangzhou Zhiyuan Electronics CO., LTD

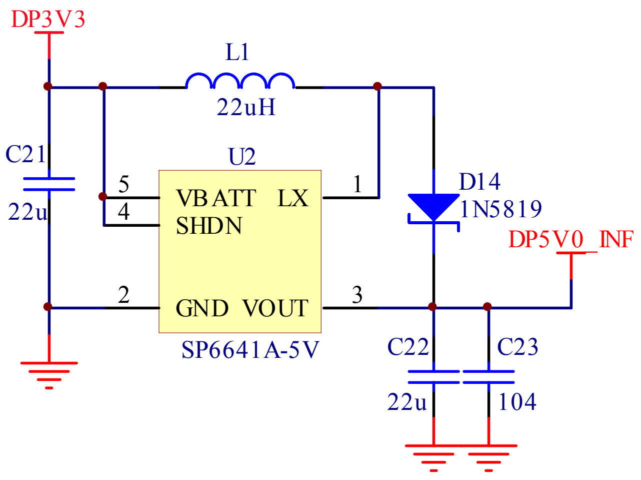

图 1.4 升压电路

1.2.4 JTAG 接口电路

电脑鼠（MicroMouse102）的 JTAG 电路采用 ARM 公司提出的标准 20 脚 JTAG 仿真调试接口，JTAG 信号的定义及与 LM3S 系列单片机的连接图如图 1.5 所示。

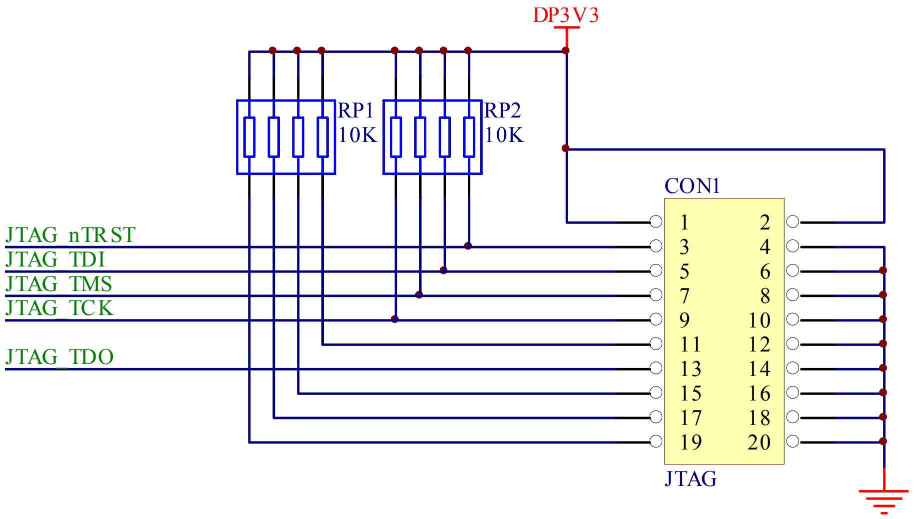

图 1.5 JTAG 接口电路

1.2.5 LED 电路

电脑鼠有 5 个独立的 LED,通过 LM3S 系统单片机的 GPIO 口直接控制，如图 1.6 所示。电路采用了 I/O 口灌电流的驱动方式来驱动 LED，LM3S 系统单片机的灌电流为 2～8mA（可配置），所以不需要驱动就可以点亮 LED。GPIO 引脚输出高电平时 LED 熄灭，低电平时 LED 点亮。

©2007 Guangzhou Zhiyuan Electronics CO., LTD

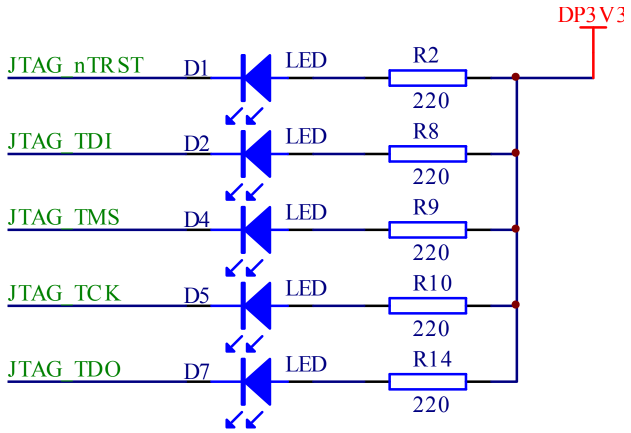

图 1.6　LED 电路

1.2.6 独立按键电路

电脑鼠设计有 2 路独立的输入按键，都直接输入到 CPU 的 GPIO 输入引脚，如图 1.7 所示。当按键未按下时，由于 R3、R4 上拉电阻的作用，CPU 检测到引脚为高电平；当按键按下时，CPU 检测到引脚为低电平。

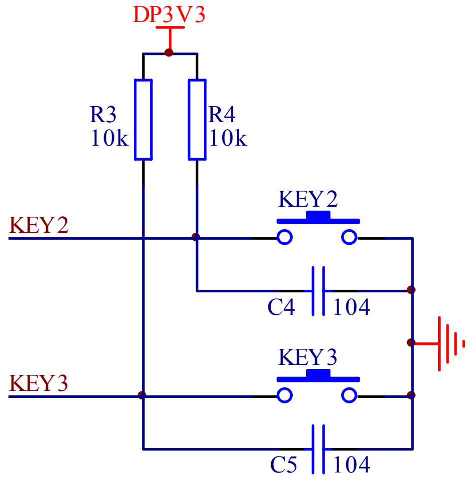

图 1.7　独立按键电路

1.2.7 电机驱动电路

电机采用直流减速电机，最高输出转速为 800 转/分钟，工作电压为 DC3V。电机驱动电路采用专用的单相直流电动机桥式驱动芯片，如图 1.8 所示。

©2007 Guangzhou Zhiyuan Electronics CO., LTD

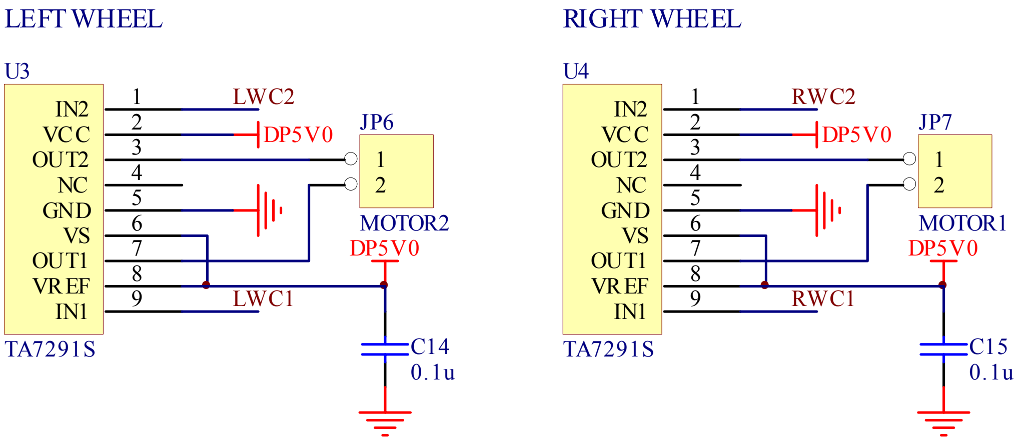

图 1.8 电机驱动电路

TA7291S 是 TOSHIBA 公司生产的单相直流电动机桥式驱动芯片，工作电压 4~20 伏，最大输出电流 400mA。电动机驱动由输入端 IN1 和 IN2 控制，控制方法如表 1.1 所示。

表 1.1 电机控制

IN1	IN2	OUT1	OUT2	电动机工作模式
0	0	∞	∞	停止，待机（STOP）
1	0	H	L	正转（CW）
0	1	L	H	反转（CCW）
1	1	L	L	刹车（BREAK）

注：电机的正转和反转与 OUT1 和 OUT2 与直流电机的接线有关，这里主要是相对的。

1.2.8 车速检测电路

车速检测用于检测并记录车体运行的路径，通过车速检测记录车体做迷宫的坐标，同时也起到控制车速和保持左右双轮的速度一致。

检测原理：在左轮和右轮的内则都贴有的光电码盘，码盘由两种颜色组成白色和黑色。红外发射管安装在车轮光电检测码盘的检测区域，当红外发射与接收管正对着黑色边时，红外线没有被反射，接收管的电阻很大；当红外发射与接收管正对着白色边时，红外线被反射，接收管的电阻很小。检测电路如图 1.9 所示。

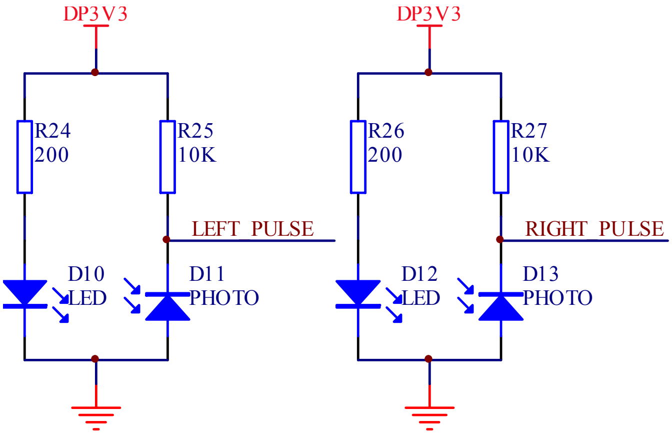

图 1.9 车速检测电路

©2007 Guangzhou Zhiyuan Electronics CO., LTD

在图 1.9 的检测电路中，红外发射与接收管正对着黑色边时，PULSE 输出高电平；正对着白色边时，PULSE 输出低电平；从黑色边到白色边，PULSE 输出一个下降沿信号；从白色边到黑色边，PULSE 则输出一个上升沿信号。

LM3S102 单片机可以检测输出脉冲的下降沿信号判断车轮转到的角速度，当检测到 13 个下降沿信号时，轮子转动了一圈。

1.2.9 红外检测电路

红外检测电路是用于迷宫挡板的检测，分为左侧、右侧、前方三个方向，三个方向的检测原理相同，某一个方向的检测电路如图 1.10 所示。

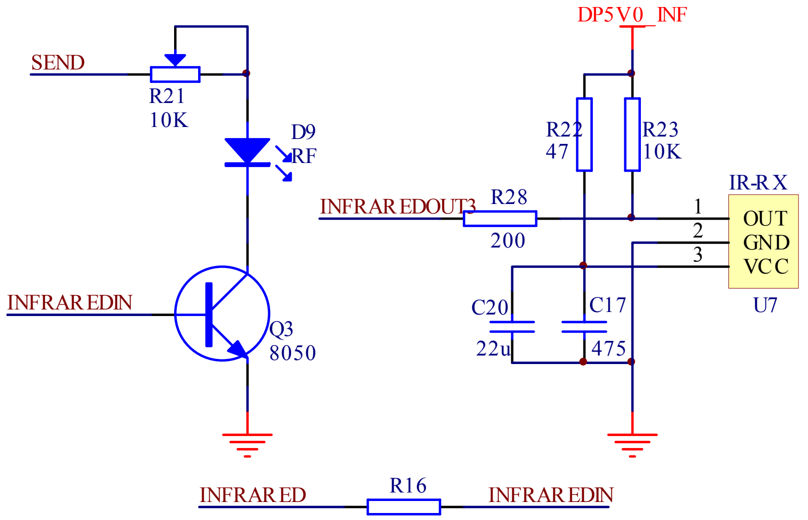

图 1.10 红外检测电路

U7（IR-RX）为一体式红外线接收传感器，当连续收到 38KHz（INFRAREDIN）的红外线信号时，将产生脉宽 10ms 左右的低电平。如果没有收到信号，便立即输出高电平。

Send 为发射控制端，高电平时发射 38KHz 的红外信号。Out（INFRAREDOUT3）为接收输出端，低电平表示收到信号。

R16 为发射端的限流电阻，起保护 Q3 管作用；R28 为接收端的限流电阻，防止过大的 I/O 端口电压，起保护 LM3S 系统单片机的 I/O 口作用。

1.2.10 CPU 及晶振电路

电脑鼠的单片机、晶体振荡器和 LDO 输出原理如图 1.11 所示。该单片机选用 LM3S102 微处理器。

©2007 Guangzhou Zhiyuan Electronics CO., LTD

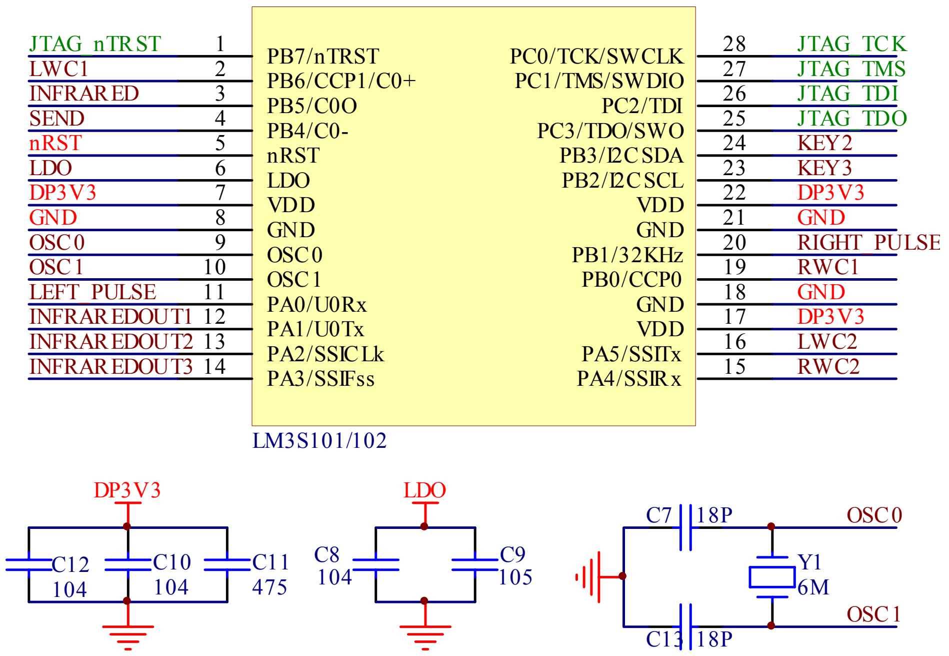

图 1.11　CPU 及晶振电路

LM3S102 微处理器有 3 组电源输入引脚，用于多点输入和多点接地，这样提高了微处理器的抗干扰能力。对于每一组电源都接了 0.1uF 的高频电容，起到高频旁路作用；接一个 475 的电容是为了使 DP3V3 电源更稳定。

LDO 输入引脚接了一个 0.1uF 的高频旁路电容和 1uF 的滤波电容，可输出稳点的 2.5～2.75V 电压。

晶体振荡器标配 6.000MHz 的石英晶振，晶振引脚是经过连接器与微处理器连接，可方便地更换晶振。

©2007 Guangzhou Zhiyuan Electronics CO., LTD

2. 反射式红外线传感器

2.1 反射式红外线传感器设计方案

反射式红外传感器探测障碍物的原理参见图 2.1 所示，发射管与接收管平行同向安装，接收管只能接收到发射回来的红外线信号。发射管向外发射红外线后，如果在有效距离范围内存在反射表面，则反射回来的信号将会被检测到。如果不存在反射表面或则反射表面距离太远，接收传感器将不能检测到信号。

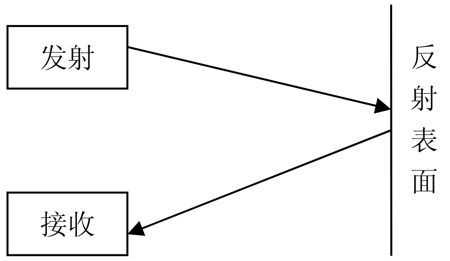

图 2.1 反射式红外线传感器探测原理

方案一：不调制的反射式红外发射-接收器。由直流电直接驱动红外管发光，这样电路简单，成本较低。但由于所有物体只要温度高于零摄氏度，都会向外发送红外线，且太阳光和日光灯中最强，所以该很容易受到外界干扰。驱动电路如图 2.2 所示。

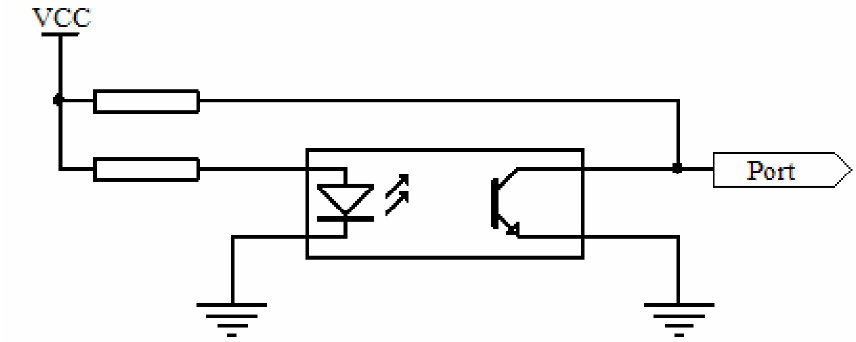

图 2.2 不调制的反射式红外发射-接收电路

方案二：脉冲调制的反射式红外发射-接收传感器。考虑到环境干扰主要是直流分量，如果采用带有交流分量的调制信号，在平均电流不变的情况下，瞬时电流可以很大，这样也大大提高了信噪比，可以有效避免外界环境变化对系统检测精度的影响。

电路原理图如图 2.3 所示，由可调电阻 R1，红外线发光管 D1 和三极管 Q1 构成的电路为红外线发射电路。R1 可以调节红外线发光管的发光强度，Q1 起驱动作用。

在接收电路中，U1 为一体式红外线接收传感器 IRM8601S,，它内部集成自动增益控制电路、带通滤波电路、解码电路及输出驱动电路。但由于它是开漏输出，所以输出端需接一个上拉电阻，见图 2.3 中的 R3。其中 R2 是限流电阻，C1 滤出电源高频干扰。

©2007 Guangzhou Zhiyuan Electronics CO., LTD

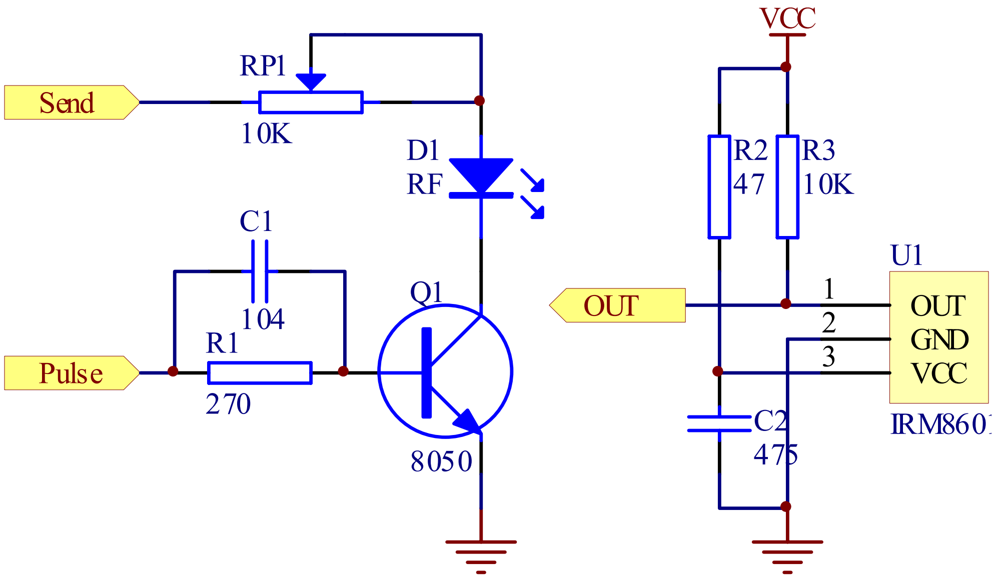

图 2.3　电路原理图

2.2 一体化红外接收头工作原理

一体式红外线接收传感器 IRM8601S，它内部集成自动增益控制电路、带通滤波电路、解码电路及输出驱动电路。当连续收到 38KHz 的红外线信号时，将产生脉宽 10ms 左右的低电平。如果没有收到信号，便立即输出高电平。如图 2.4 所示，Send 为发射控制端，高电平时发射 38KHz 的红外信号。Out 为接收输出端，低电平表示收到信号。

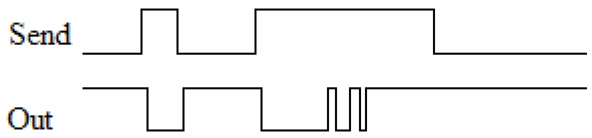

图 2.4　波形图

2.3 检测障碍物的软件设计

根据接收头是否检测到经过反射的红外线信号，就可以判断是否存在障碍物。由于接收头检测到信号时只产生一个负脉冲，所以只需要在检测时使能红外线发射，一次检测结束后使能无效，程序设计参考流程图如图 2.5 所示。

©2007 Guangzhou Zhiyuan Electronics CO., LTD

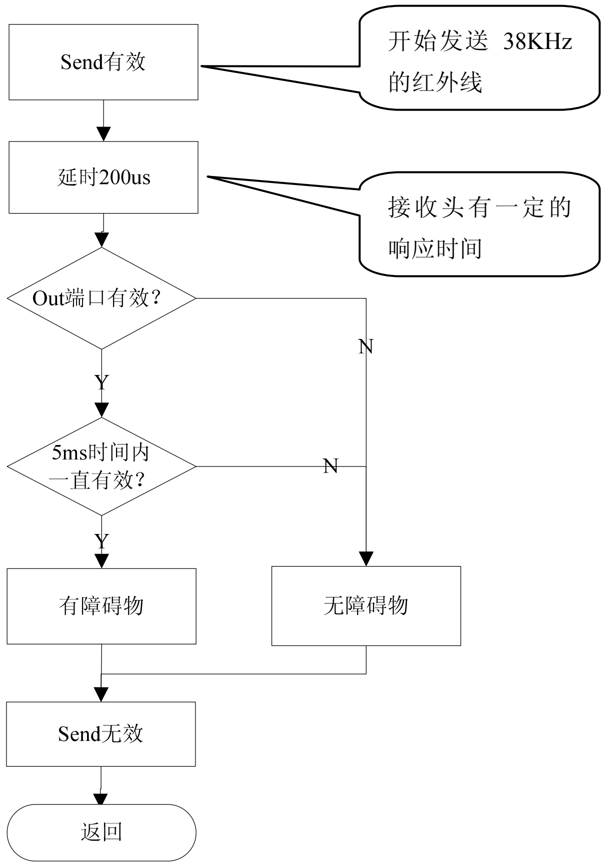

图 2.5 程序流程图

©2007 Guangzhou Zhiyuan Electronics CO., LTD

3. 迷宫挡板检测

3.1 原理分析

Micromouse 中文名为“电脑鼠”，电脑鼠在迷宫中行进时是靠侦测路面情况前进的。它的左右传感器不但要检测是否存在支路（没有挡板就是一条支路）还要避免和挡板碰触。因此电脑鼠每一侧在正常情况就需要两组红外传感器，一组检测稍微远一点的距离，判断是否存在支路，一组检测稍微近一点的距离，判断是否即将碰触挡板。

由此看出，电脑鼠每侧都需要两组传感器，但是如果只用一组传感器来完成两个参数的检测是否可行呢？我们知道，如果使用非调制的普通红外接收头，就可以根据接收到的信号的强弱来计算距离，可是非调制的抗干扰差，但是如果使用调制的一体化接收头，检测信号输出的是数字信号，这样通过检测传感器输出信号的强弱来计算距离的方法肯定行不通，但还有没有其他方法可以实现距离的测量呢？答案是肯定的。

把上面测距的原理反过来，我们可以通过改变发射出的接收传感器能够识别的信号强度，当接收头刚好能接收到信号时，记录下此时发射的强度，这样也就可以大致测算出距离。

改变输出接收传感器所能识别信号的强度的方法有两种：

1. 改变输出信号的能量，改变输出信号的能量又有两种方法：
   - 改变输出信号的幅度，如图 2.3 所示，改变 Send 端的输出电压或调节可调电阻 R1 的阻值就可以实现
   - 改变输出信号的占空比，如图 2.3 所示，改变输出的 38KHz 信号的占空比就可以实现

2. 改变输出信号的频率，由于一体化接收头是 38KHz 的带通滤波器，所以发射信号的频率偏离 38KHz 越多，能检测到的有效信号就越少。这样也就可以改变有效发射信号的强度。

如果通过改变输出信号的幅度，就需要还需要一个 D/A 转换器或手动调节 R1，这样要不增加了成本要不不适用于自动控制。如果通过改变占空比的方式就需要使用 PWM 功能，但 PWM 功能已经被驱动小车的电机占用，所以只能通过改变输出信号的频率来实现测距，这样只需要一个定时器就可以完成功能。

3.2 调制信号产生

本设计中采用定时器 1 产生 38KHz 的调制信号，由 PB5 输出，该端口连接到图 2.3 中的 Pulse 端口。在中断中翻转 PB5 输出信号，所以要产生频率为 f 的脉冲，定时器的频率要为 2f。在本设计中要产生 38KHz 的频率，定时器中断频率为 76KHz。

程序清单 3.1 为定时器 1 的初始化函数，程序清单 3.2 为中断服务函数，在这里翻转 PB5 口输出状态。

程序清单 3.1 定时器 1 初始化

```
void PULSEIni(void)
{
    GPIODirModeSet(GPIO_PORTB_BASE, SEND | PULSE, GPIO_DIR_MODE_OUT);        // 设置为输
出
```


©2007 Guangzhou Zhiyuan Electronics CO., LTD

```
    GPIOPinWrite( GPIO_PORTB_BASE,SEND | PULSE,0);               // 红外线初始时停止发射
    SysCtlPeripheralEnable( SYSCTL_PERIPH_TIMER1 );              // 使能定时器 1 外设
    TimerConfigure(TIMER1_BASE, TIMER_CFG_32_BIT_PER);           // 设置定时器 1 为周期触发
    TimerLoadSet(TIMER1_BASE, TIMER_A, SysCtlClockGet()/76000);  // 设置定时器装载值
    TimerIntEnable(TIMER1_BASE, TIMER_TIMA_TIMEOUT);
    TimerEnable(TIMER1_BASE, TIMER_A);
    IntEnable(INT_TIMER1A);
}
```

程序清单 3.2 定时器 1 服务函数

```
void Timer1A_ISR(void)
{
    TimerIntClear(TIMER1_BASE, TIMER_TIMA_TIMEOUT);              // 清除定时器 1 中断
    GPIOPinWrite(GPIO_PORTB_BASE, PULSE,GPIOPinRead(GPIO_PORTB_BASE, PULSE) ^ PULSE);
                                                                 // 翻转 GPIO B5 端口
}
```

3.3 抗干扰处理

红外线在空气中传播和反射受外界的干扰，如果测量距离刚好处在能够检测到信号的临界状态，保持距离不变，传感器输出信号也可能不确定。这样就需要在软件中进行抗干扰处理。参考程序如程序清单 3.3 所示。

程序清单 3.3 抗干扰处理程序

```
GPIOPinWrite( GPIO_PORTB_BASE,SEND , SEND);        // 发送脉冲
Delay(150);                                        // 延时
for(i=0,j=0;i<10;i++)                              // 检测接收信号
{
    if(GPIOPinRead(GPIO_PORTA_BASE, OUT_L)==0)
        j++;
}
GPIOPinWrite( GPIO_PORTB_BASE,SEND , ~SEND);       // 停止发送
if(j>5)                                            // 左边存在挡板
{
    ⋮
}
else                                               // 左边存在支路
{
    ⋮
}
```

图 3.1 为抗干扰程序在 Micromouse 中运行后用逻辑分析仪抓到的波形图，Pulse 为 38KHz 的输出信号，Send 高电平有效，有效时发送红外线脉冲，OUT 为一体化接收头输

 ©2007 Guangzhou Zhiyuan Electronics CO., LTD

出端，该图所示为接收头探测到障碍物，软件在 Send 信号无效（下降沿）前完成检测 OUT 输出信号，从图中可以看出，此时正处于 OUT 有效信号的中间，所以软件里延时参数能保证正确检测到信号。

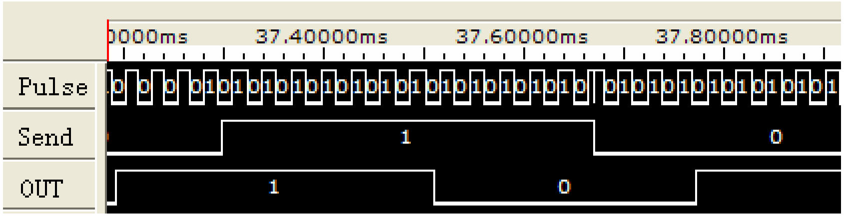

图 3.1 传感器检测波形图

3.4 软件设计参考

图 3.2 为用一组红外实现两组参数（是否存在挡板和是否太接近挡板）的检测流程图。在 Micromouse 中，用到了三组（左、前、右）反射式红外检测传感器，左边和右边的传感器各自都需要检测两组参数，而前方的传感器只需要探测有无挡板，存在挡板就必须根据策略转换行进方向，若不存在就可以继续前进。如图 3.3 所示为 Micromouse 红外检测的程序设计流程图。红外检测参考程序见程序清单 3.4 所示，该程序中使用了五个 LED 用来指示传感器检测的状态，由于这几个 LED 硬件上连接到 JTAG，关于如何切换 GPIO 和 JTAG 功能参见 6 使用 JTAG 引脚作 GPIO。

©2007 Guangzhou Zhiyuan Electronics CO., LTD

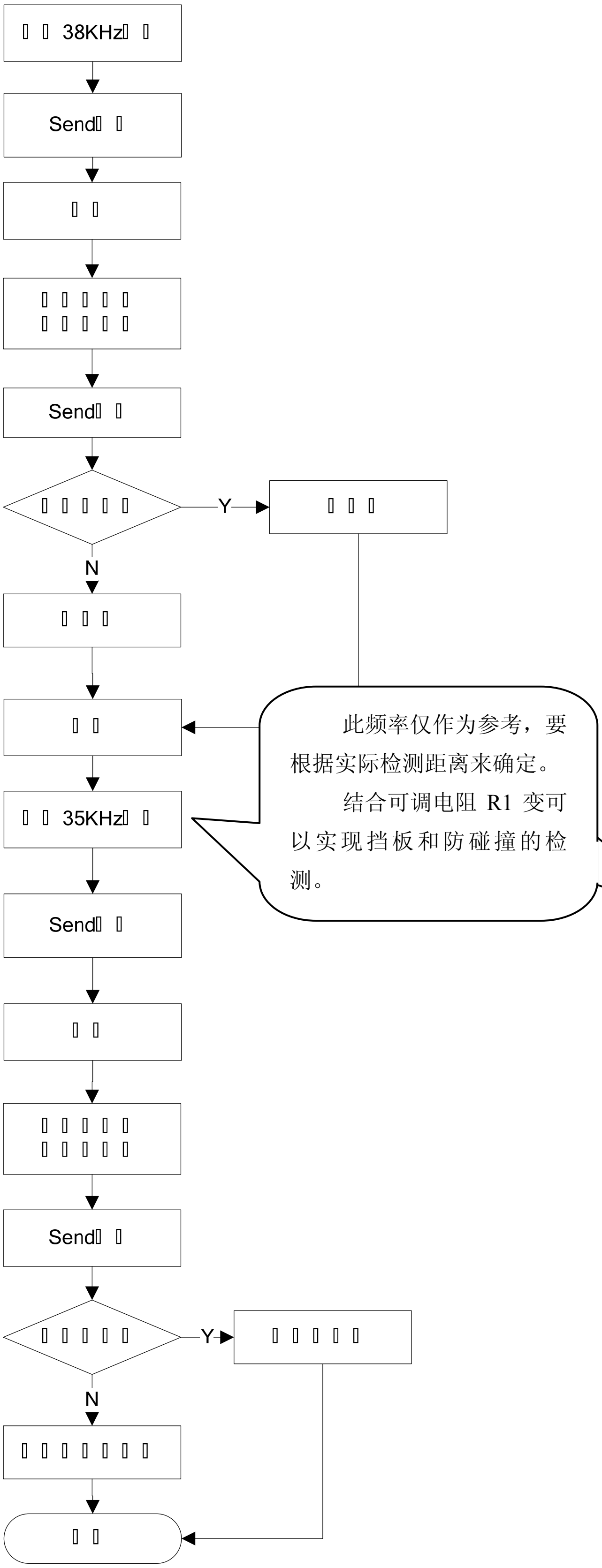

图 3.2 探测远近两个参数的流程图

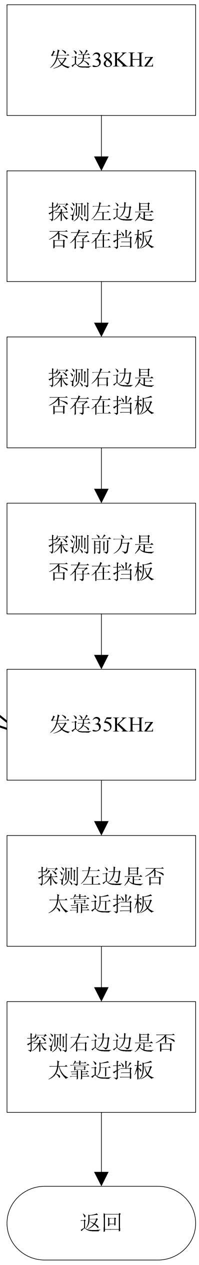

图 3.3 Micromouse 红外检测流程图

©2007 Guangzhou Zhiyuan Electronics CO., LTD

程序清单 3.4 Micromouse 红外检测函数

```
//------------------------------------------------------------------------------------------------------------------------------
--
// 函数名称: Check_Infrared
// 函数功能: 红外检测函数
// 入口参数: option，选择检测的参数。
//                                    0，检测所有参数
//                                    1，检测左侧挡板
//                                    2，检测前方挡板
//                                    3，检测右侧挡板
//                                    4，检测左侧安全距离
//                                    5，检测右侧安全距离
// 出口参数: State，反应传感器状态。State 每四位代表一个参数状态。从左至右 20 位（0xXXXXX）分别
// 表示五个参数：左侧远距，左侧近距，前方，右侧近距，右侧远距。如值为 0x11100 表示左侧有挡板，
// 且太靠近左侧挡板，前方有挡板，右方没有挡板。
//------------------------------------------------------------------------------------------------------------------------------
--
unsigned long Check_Infrared(unsigned char option)
{
    unsigned long State = 0;
    unsigned char j = 0,i = 0;
    if(option == 4 || option == 5)       // 如果只需测近距，直接跳到装载定时器为 35KHz 的程序处
        goto jinju;
    TimerLoadSet(TIMER1_BASE, TIMER_A, SysCtlClockGet()/76000);     // 设置定时器装载值
    if(option==0||option==1)
    {
        GPIOPinWrite( GPIO_PORTB_BASE,SEND , SEND);                  // 发送调制脉冲
        Delay(150);
        GPIOPinWrite( GPIO_PORTB_BASE,SEND , ~SEND);                 // 停止发送调制脉冲
        for(i=0,j=0;i<10;i++)                                        // 传感器输出信号检测
        {
            if(GPIOPinRead(GPIO_PORTA_BASE, OUT_L)==0)
                j++;
        }
        if(j>5)                                                      // 左边存在挡板
        {
            if(option == 1)                                          // 只测此一个参数
                return 1;                                            // 直接返回
            GPIOPinWrite(GPIO_PORTC_BASE, DLL, 0);                   // 点亮指示灯 DLL
            State |= 1 << 16;
        }
        else                                                         // 左边存在支路
        {
```


©2007 Guangzhou Zhiyuan Electronics CO., LTD

```
                if(option == 1)
                        return 0;
                GPIOPinWrite(GPIO_PORTC_BASE, DLL, DLL);            //  熄灭指示灯 DLL
                State |= 0 << 16;
        }
}
if(option==0||option==2)
{
        GPIOPinWrite( GPIO_PORTB_BASE,SEND , SEND);
        Delay(150);
        GPIOPinWrite( GPIO_PORTB_BASE,SEND , ~SEND);
        for(i=0,j=0;i<10;i++)
        {
                if(GPIOPinRead(GPIO_PORTA_BASE, OUT_F)==0)
                        j++;
        }
        if(j>5)                                                     //  前面存在挡板
        {
                if(option == 2)
                        return 1;
                GPIOPinWrite(GPIO_PORTC_BASE, DF, 0);               //  点亮指示灯 DF
                State |= 1 << 8;
        }
        else                                                        //  前面没有挡板
        {
                if(option == 2)
                        return 0;
                GPIOPinWrite(GPIO_PORTC_BASE, DF, DF);              //  熄灭指示灯 DF
                State |= 0 << 8;
        }
}
if(option==0||option==3)
{
        GPIOPinWrite( GPIO_PORTB_BASE,SEND , SEND);
        Delay(150);
        GPIOPinWrite( GPIO_PORTB_BASE,SEND , ~SEND);
        for(i=0,j=0;i<10;i++)
        {
                if(GPIOPinRead(GPIO_PORTA_BASE, OUT_R)==0)
                        j++;
        }
        if(j>5)                                                     //  右边存在挡板
        {
                if(option == 3)
```

 ©2007 Guangzhou Zhiyuan Electronics CO., LTD

```
                return 1;
            GPIOPinWrite(GPIO_PORTB_BASE, DRR, 0);              // 点亮指示灯 DRR
            State |= 1 << 0;
        }
        else                                                    // 右边存在支路
        {
            if(option == 3)
                return 0;
            GPIOPinWrite(GPIO_PORTB_BASE, DRR, DRR);            // 熄灭指示灯 DRR
            State |= 0 << 0;
        }
    }
jinju:
    TimerLoadSet(TIMER1_BASE, TIMER_A, SysCtlClockGet()/70000);    // 设置定时器装载值
    if(option == 0||option == 4)
    {
         GPIOPinWrite( GPIO_PORTB_BASE,SEND , SEND);
        Delay(150);
        GPIOPinWrite( GPIO_PORTB_BASE,SEND , ~SEND);
        for(i=0,j=0;i<10;i++)
        {
            if(GPIOPinRead(GPIO_PORTA_BASE, OUT_L)==0)
                j++;
        }
        if(j>8)                                                 // 左边太靠近挡板,应向右调
整
        {
            if(option == 4)
                return 1;
            GPIOPinWrite(GPIO_PORTC_BASE, DL, 0);               // 点亮指示灯 DL
            State |= 1 << 12;
        }
        else                                                    // 左边正常
        {
            if(option == 4)
                return 0;
            GPIOPinWrite(GPIO_PORTC_BASE, DL, DL);              // 熄灭指示灯 DL
            State |= 0 << 12;
        }
    }
    if(option ==0 || option == 5)
    {
        GPIOPinWrite( GPIO_PORTB_BASE,SEND , SEND);
        Delay(150);
```


©2007 Guangzhou Zhiyuan Electronics CO., LTD

```
            GPIOPinWrite( GPIO_PORTB_BASE,SEND , ~SEND);
            for(i=0,j=0;i<10;i++)
            {
                  if(GPIOPinRead(GPIO_PORTA_BASE, OUT_R)==0)
                        j++;
            }
            if(j>8)                                                      // 右边太靠近挡板,应向左调
整
            {
                  if(option == 5)
                        return 1;
                  GPIOPinWrite(GPIO_PORTC_BASE, DR, 0);                  // 点亮指示灯 DR
                  State |= 1 << 4;
            }
            else                                                         // 右边正常
            {
                  if(option == 5)
                        return 0;
                  GPIOPinWrite(GPIO_PORTC_BASE, DR, DR);                 // 熄灭指示灯 DR
                  State |= 0 << 4;
            }
      }
      return(State);                                                     // 返回
}
```


©2007 Guangzhou Zhiyuan Electronics CO., LTD

4. 电机的调速

4.1 电机的调速

直流电机的转速控制在本设计中通过PWM来控制，LM3S102单片机则刚有两路PWM输出，非常适合用于控制两个电机的转速。

两路PWM是LM3S102通用定时器0（Timer0）的三种工作模式之一，16位PWM模式。该模式是将一个32位的定时器，折分成两个16位的定时器TimerA和TimerB。这些定时器为计数寄存器(GPTMTnR)递减计数，递减到0时自动加载预装载值(GPTMTnILR)。当然预装载值也是由用户设定，该直也就决定了定时周期，也即PWM的输出周期。

当计数器的值与预装载值相等时，输出PWM信号有效，当计数器的值与匹配寄存器（GPTMnMATCHR）的值相等时，输出PWM信号失效。通过软件可以设定PWM输的信号有效和信号无效的电平状态。当GPTMCTL寄存器的TnPWML位值为0时，信号有效为高电平，信号无效为低电平；TnPWML位值为1时，则反之。如图 4.1所示。

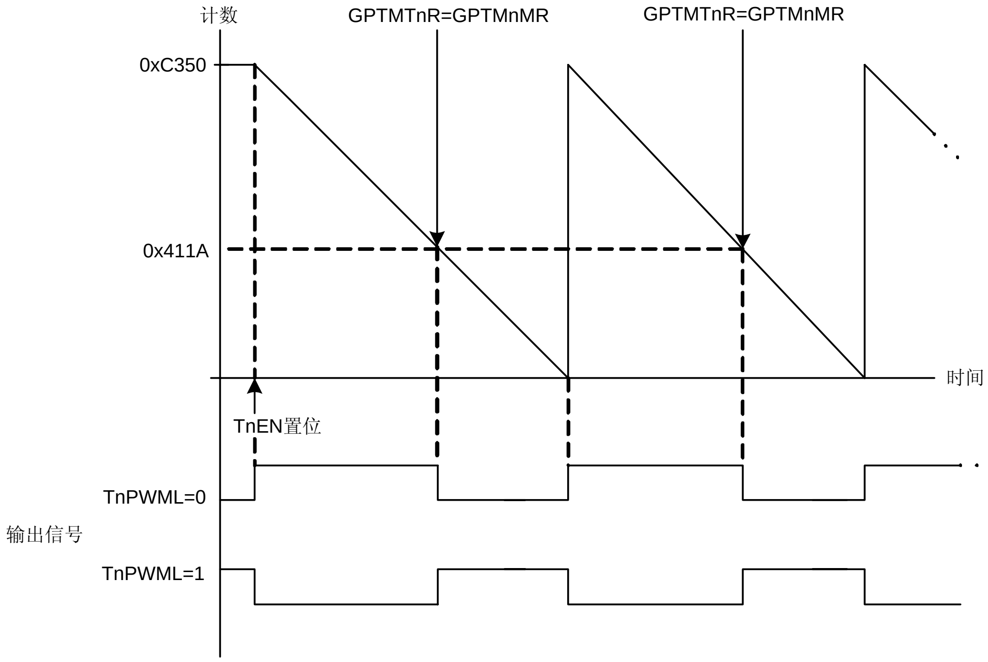

图 4.1 16位PWM模式输出

占空比的约定：占空比为在一个周期内，输出有信号有效电平占整个周期时间的比率。在这里为以统一软件控制的约定，用户API函数输入的占空比值越大，电机转速越快，正向运行和反向运行都一样。

为了简化占空比输出的计算，将计数寄存器与匹配寄存器值相等时，输出的电平信号为驱动电机的有效信号。例如将PWM周期时间设定为60000个时钟节拍，需要输出驱动电机的占空比为75%，则设置匹配寄存器值为75*6000。

由于电机的转向不一样，所以电机驱动的有效电平也需要调整，通过控制 TnPWML实现。

©2007 Guangzhou Zhiyuan Electronics CO., LTD

4.2 程序设计

Timer0 的两路 16 定时器 TimerA 和 TimerB 的 PWM 输出引脚分别为 PB0 和 PB6，PB0 和 PB6 分别控制左轮和右轮驱动器 TA7291S 的 IN1 引脚，而它们的 IN2 引脚分别由 GPIO 输出的 PA4 和 PA5 控制。

左轮的控制函数如程序清单 4.1 所示。

该函数的第 1 个参数 sel 为选择轮子的控制方式：0 为停止，1 为轮子向前，2 为轮子向后；percen 参数为占空比，其最大值为 99，最小值为 1，对于轮子的停止控制该参数无效。

程序清单 4.1 左轮控制函数

```
void LeftWheelRun(int sel,unsigned char percen)
{
    switch(sel)
    {
        /*轮子停止转动*/
        case 0:
            TimerDisable(TIMER0_BASE,TIMER_A);                              // 禁止定时器
            GPIOPinWrite(GPIO_PORTA_BASE,LWC2,0xff);                        // 控制引脚输出高电平
            GPIODirModeSet(GPIO_PORTB_BASE, LWC1, GPIO_DIR_MODE_OUT);       // GPIO 输出
            GPIOPinWrite(GPIO_PORTB_BASE,LWC1, 0xff);                       // GPIO 输出高电平
            break;
        /*左轮向前*/
        case 1:
            GPIOPinWrite(GPIO_PORTA_BASE,LWC2, 0xff);                       // PA4 输出高电平
            TimerControlLevel(TIMER0_BASE,TIMER_A,true);                    // PWM 有效电平方向
            GPIODirModeSet(GPIO_PORTB_BASE, LWC1, GPIO_DIR_MODE_HW);        // PWM 输出

            TimerMatchSet(TIMER0_BASE,TIMER_A,percen*600);                  // 设置占空比
            TimerEnable(TIMER0_BASE,TIMER_A);                               // 使能定时器
            break;
        /*左轮向后*/
        case 2:
            GPIOPinWrite(GPIO_PORTA_BASE,LWC2, 0);                          // PA4 输出低电平
            TimerControlLevel(TIMER0_BASE,TIMER_A,false);                   // PWM 有效电平方向
            GPIODirModeSet(GPIO_PORTB_BASE, LWC1, GPIO_DIR_MODE_HW);        // PWM 输出
            TimerMatchSet(TIMER0_BASE,TIMER_A,percen*600);                  // 设置占空比
            TimerEnable(TIMER0_BASE,TIMER_A);                               // 使能定时器
            break;
    }
```


©2007 Guangzhou Zhiyuan Electronics CO., LTD

```
}
```

右轮的控制函数如程序清单 4.2 所示。

该函数的第 1 个参数 sel 为选择轮子的控制方式：0 为停止，1 为轮子向前，2 为轮子向后；percen 参数为占空比，其最大值为 99，最小值为 1，对于轮子的停止控制该参数无效。

程序清单 4.2 右轮控制函数

```
void RightWheelRun(int sel,unsigned char percen)
{
    switch(sel)
    {
        /*轮子停止转动*/
        case 0:
            TimerDisable(TIMER0_BASE,TIMER_B);                    // 禁止定时器
            GPIOPinWrite(GPIO_PORTA_BASE,RWC2,0xff);              // 控制引脚输出高电平
            GPIODirModeSet(GPIO_PORTB_BASE, RWC1, GPIO_DIR_MODE_OUT);  // GPIO 输出
            GPIOPinWrite(GPIO_PORTB_BASE,RWC1,0xff);              // GPIO 输出高电平
            break;
        /*右轮向后*/
        case 2:
            GPIOPinWrite(GPIO_PORTA_BASE,RWC2, 0xff);             // PA4 输出高电平
            TimerControlLevel(TIMER0_BASE,TIMER_B,true);          // PWM 有效电平方向
            GPIODirModeSet(GPIO_PORTB_BASE, RWC1, GPIO_DIR_MODE_HW);   // PWM 输出

            TimerMatchSet(TIMER0_BASE,TIMER_B,percen*600);        // 设置占空比
            TimerEnable(TIMER0_BASE,TIMER_B);                     // 使能定时器
            break;
        /*右轮向前*/
        case 1:
            GPIOPinWrite(GPIO_PORTA_BASE,RWC2, 0);                // PA4 输出低电平
            TimerControlLevel(TIMER0_BASE,TIMER_B,false);         // PWM 有效电平方向
            GPIODirModeSet(GPIO_PORTB_BASE, RWC1, GPIO_DIR_MODE_HW);   // PWM 输出
            TimerMatchSet(TIMER0_BASE,TIMER_B,percen*600);        // 设置占空比
            TimerEnable(TIMER0_BASE,TIMER_B);                     // 使能定时器
            break;
    }
}
```

需要注意的是，当 PWM 信号禁止后，其输出引脚的电平状态是保持静止时的状态（可能为低电平也可能为高电平），导致电机可能不能停止，所以在制停电机时，需要将 PWM 引脚改为 GPIO 输出，并且出高电平，使电机刹车停止。

定时器 PWM 初始化函数如程序清单 4.3 所示。

程序清单 4.3 定时器 PWM 初始化

©2007 Guangzhou Zhiyuan Electronics CO., LTD

```
void PWMTimer0AIni(void)
{
	SysCtlPeripheralEnable(SYSCTL_PERIPH_TIMER0);                    // 使能定时器 0 模
块
	GPIODirModeSet(GPIO_PORTB_BASE, LWC1|RWC1 , GPIO_DIR_MODE_OUT);

	/* 控制引脚输出*/
	GPIOPinWrite(GPIO_PORTB_BASE, LWC1|RWC1 , 0xff);                  // GPIO 输出高电
平
	GPIODirModeSet(GPIO_PORTA_BASE, LWC2|RWC2 , GPIO_DIR_MODE_OUT); // 控制引脚输出
	GPIOPinWrite(GPIO_PORTA_BASE, LWC2|RWC2 , 0xff);                  // GPIO 输出高电
平

	/* 定时器配置*/
	TimerConfigure(TIMER0_BASE,TIMER_CFG_16_BIT_PAIR
	                      |TIMER_CFG_A_PWM|TIMER_CFG_B_PWM);         // 16 位 PWM 输出
	TimerControlLevel(TIMER0_BASE,TIMER_A,false);                      //有效信号为低电
平
	TimerControlLevel(TIMER0_BASE,TIMER_B,false);
	TimerLoadSet(TIMER0_BASE,TIMER_A,60000);                           // 设定 PWM 频率
	TimerLoadSet(TIMER0_BASE,TIMER_B,60000);
}
```


©2007 Guangzhou Zhiyuan Electronics CO., LTD

5. Micromouse 车速检测

5.1 车速检测程序设计

本设计中选用了 LM3S102 的 PA0 和 PB1 分别检测左轮和右轮的下降沿的脉冲个数，为了快速向应检测信号，使用了下降沿触发中断。LM3S102 单片机的特点，任何一个 GPIO 引脚都可以配置为中断输入，并且可以作任意设定为高电平触发、低电平触发、下降沿触发、上升沿触发和上升或下降沿触发 5 种模式。本应用中使用下降沿触发，其初始化如程序清单 5.1 所示。

程序清单 5.1 轮子脉冲检测初始化

```
void WheelPulseIni(void)
{
        // 配置引脚为输入
        GPIODirModeSet(GPIO_PORTA_BASE, PULSE_R, GPIO_DIR_MODE_IN);
        GPIODirModeSet(GPIO_PORTB_BASE, PULSE_L, GPIO_DIR_MODE_IN);
       // 配置引脚下降沿触发中断
        GPIOIntTypeSet(GPIO_PORTA_BASE,PULSE_R,GPIO_FALLING_EDGE);
        GPIOIntTypeSet(GPIO_PORTB_BASE,PULSE_L,GPIO_FALLING_EDGE);
        // 使能引脚输入中断
        GPIOPinIntEnable(GPIO_PORTA_BASE,PULSE_R);
        GPIOPinIntEnable(GPIO_PORTB_BASE,PULSE_L);
        // 使能 GPIO PA 口和 GPIO PB 口中断
        IntEnable(INT_GPIOA);
        IntEnable(INT_GPIOB);
}
```

左右轮检测脉冲中断处函数如程序清单 5.2 所示。

程序清单 5.2 左右轮检测脉冲中断处函数

```
//----------------------------------------------------------------------------------
// 函数名称: GPIO_Port_A_ISR
// 函数功能: 右轮检测脉冲中断处函数
//----------------------------------------------------------------------------------
void GPIO_Port_A_ISR (void)
{
        unsigned char    IntStatus;
        IntStatus = GPIOPinIntStatus(GPIO_PORTA_BASE,true);              // 读 PA 口中断状态
        if(IntStatus&PULSE_R)                                            // 是否为左轮脉冲中断
        {
                PulCount_R++;
                if(PulCount_R >= RightPulse)
                {
                        RightWheelRun(0, 1);
```

 ©2007 Guangzhou Zhiyuan Electronics CO., LTD

```
                WheelStop_R= 1;
            }
            GPIOPinIntClear(GPIO_PORTA_BASE,PULSE_R);             // 清中断
        }
}
//---------------------------------------------------------------------------------------
// 函数名称: GPIO_Port_B_ISR
// 函数功能: 左轮检测脉冲中断处函数
//---------------------------------------------------------------------------------------
void GPIO_Port_B_ISR (void)
{
        unsigned char IntStatus;
        IntStatus = GPIOPinIntStatus(GPIO_PORTB_BASE,true);            // 读 PA 口中断状态
        if(IntStatus&PULSE_L)                                           // 是否为右轮脉冲中断
        {
            PulCount_L++;
            if(PulCount_L>= LeftPulse)
            {
                WheelStop_L= 1;
                LeftWheelRun(0, 1);
            }
            GPIOPinIntClear(GPIO_PORTB_BASE,PULSE_L);             // 清中断
        }
}
```


©2007 Guangzhou Zhiyuan Electronics CO., LTD

6. 使用 JTAG 引脚作 GPIO

由于 LM3S102 管脚资源有限，我们需要使用 PB7 和 PC0~3 做 GPIO 使用，并且可以从 GPIO 功能转换为 JTAG 引脚功能。该五个引脚都分别与 LED 相连，LED 可以在程序调试时做指示灯使用。需要特别注意的是，该五个引脚若无法恢复 JTAG 功能，则该芯片无法再次编程。

6.1 使用方法

使用 GPIODirModeSet()很轻易地将 PB7 和 PV3~0 引脚设置为 GPIO 的输入/输出和 JTAG 功能，系统复位后这几个引脚恢复默认的 JTAG 功能，所以程序只需要解决怎样使这些引脚在 GPIO 和 JTAG 功能间互换。

在程序清单 6.1 中，是通过上电或复位后首先检测 KEY2 引脚，如果按键有效则进入死循环，这些引脚则保持 JTAG 功能。否则这些引脚被设置为 GPIO 功能，并继续执行主程序。

所以要使这些引脚作为 JTAG 功能，首先按住 KEY2 键，然后复位。要使 MCU 复位后设置这些引脚为 GPIO 口且可以继续执行 flash 内的程序，则在复位时放开 KEY2。

程序清单 6.1 JTAG 引脚作 GPIO

```
//PB
#define KEY2                    GPIO_PIN_3
#define DRR                     GPIO_PIN_7
//PC
#define DLL                     GPIO_PIN_3
#define DL                      GPIO_PIN_0
#define DF                      GPIO_PIN_1
#define DR                      GPIO_PIN_2
main()
{
SysCtlPeripheralEnable( SYSCTL_PERIPH_GPIOB );          // 使能 GPIO B 口外设
SysCtlPeripheralEnable( SYSCTL_PERIPH_GPIOC );          // 使能 GPIO C 口外设

GPIODirModeSet(GPIO_PORTB_BASE, KEY2, GPIO_DIR_MODE_IN);
if(GPIOPinRead(GPIO_PORTB_BASE, KEY2)==0)               // 如果按下 KEY2 就进入
JTAG
{
    while(1);
}
/* 把五个 JTAG 口设置为 GPIO 输出口 */
GPIODirModeSet(GPIO_PORTC_BASE, DLL | DL | DF | DR, GPIO_DIR_MODE_OUT);
GPIODirModeSet(GPIO_PORTB_BASE, DRR , GPIO_DIR_MODE_OUT);
```


©2007 Guangzhou Zhiyuan Electronics CO., LTD

```
      ⋮

while(1)
{
      ⋮

}
}
```

 ©2007 Guangzhou Zhiyuan Electronics CO., LTD